UNIT – 4 Creating Basic App

What is Dalvik Virtual Machine?

Firstly let us understand what a virtual machine is? It is basically a software implementation of a physical computer. This implementation works like a real physical computer. It even compiles and runs programs the same as a physical computer. It can be understood like an emulator. There are some issues with virtual machines too. One is that it is less efficient when compared to physical computers. Another issue is its performance, which is unstable when multiple virtual machines are working simultaneously on the same machine.

Even though Java Virtual machine has a high performance and provides great memory management, it is not optimized for low-powered devices. Dalvik VM is also a virtual machine that is highly optimized for mobile devices. Thus, it provides all the three things, that are **memory management, high performance as well as battery life**. It is strictly developed for Android mobile phones.

Role of the Dalvik Virtual Machine

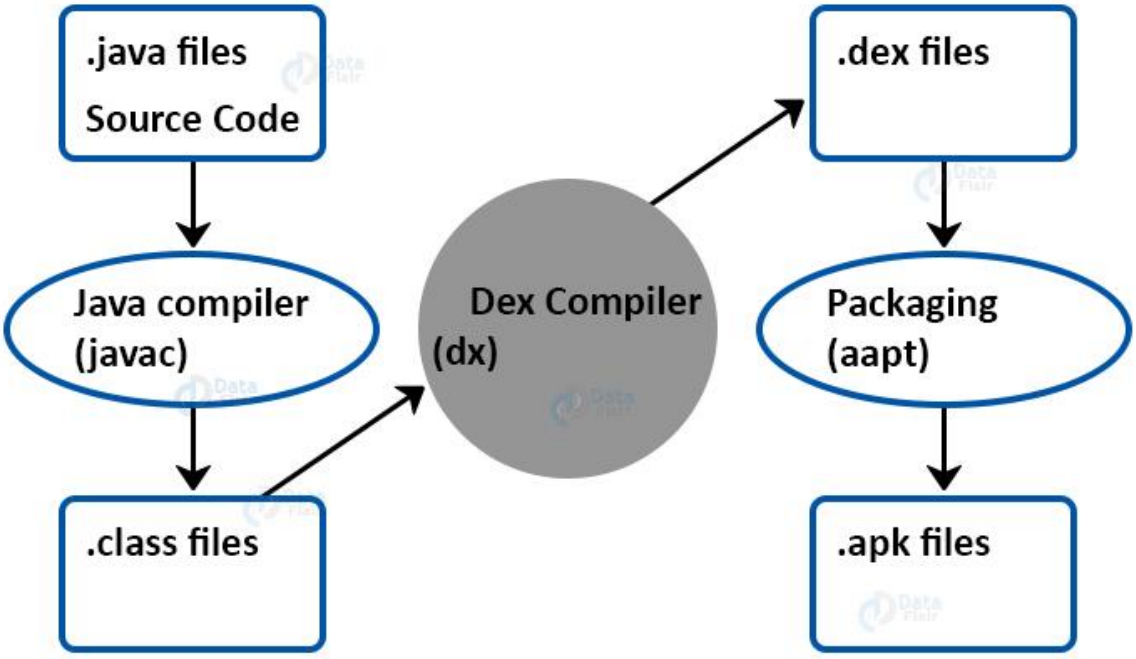
The Role of the DVM in Android includes:

- Optimizing the Virtual Machine for memory, battery life, and performance
- Conversion of class files into .dex file through Dex compiler that runs on Dalvik VM.
- Converting multiple class files into dex files.

The Dex compiler helps convert the class file into .dex file, the following image shows how it flows:

- First of all the .java file converts into .class file with the help of Java compiler.
- Next .class file converts into .dex file using Dec compiler.
- Then finally the packaging process is handled by the Android Assets packaging (aapt) tools.

Working of Dex Compiler



Dalvik Virtual Machine vs Android Runtime:

Let us see comparison between Android DVM and Android Runtime:

Dalvik Virtual Machine	Android Runtime
Dalvik is slower in comparison to Android Runtime	Android Runtime is faster than Dalvik Virtual Machine
Dalvik Virtual Mchine takes less time to boot, Booting is fast	Android Runtime takes more time to boot, Booting is slow
The Cache builds up fast over time, reducing the reboot time	The cache is built at first boot, increasing the reboot time
Dalvik Virtual Mchine needs less space as it uses Just In Time compiler	Android Runtime needs more space as it uses AOT.
Dalvik Virtual Mchine utilizes more battery, thus it has low battery performance	Android Runtime utilizes less battery, thus it has high battery performance
Dalvik Virtual Mchine has a poor Garbage collection when compared to Android Runtime	Android Runtime has a better Garbage collection when compared to DVM
In Android DVM, apps are less responsive in accordance with Android Runtime	Apps here are very responsive and work smoothly.
Dalvik Runtime Virtual Machine converts bytecode every time the application launches	On the other hand, Android Runtime converts the bytecode only once at the time of installation of application
It is a stable and time-tested virtual machine	It is highly experimented and new
DVM is the choice of Android developers	It doesn't have a lot of support from app developers until now
DVM works better for lower internal storage devices as space occupied is less	It consumes more internal storage space, as it stores compiled apps in addition to the APKs
It came prior to Android Runtime and it is replaced with Android Runtime	Android Runtime is the upgraded version of Dalvik Virtual Machine and comes up with a lot of improvements

What is Android Manifest File – androidmanifest.xml file?

As we know that every android project that we make, need to have an Android Manifest file within. This file is present in the root of the project source set. The reason why it is a must to have it is that it describes all the important information of the application to the Android build tools.

Android Manifest File

The manifest file in Android is generally created automatically as soon as the app is built in Android Studio.

Structure of a Manifest file in Android is:

```
<manifest>
   <application>
<activity android:name="com.example.applicationname.MainActivity" >
       </activity>
   </application>
</manifest>
```

The information that is stored in the Manifest file is as follows:

- The name of the application's package, it is generally the code's namespace. This information is used to determine the location of the code while building the project.
- Another component is the one, that includes all the activities, services, receivers, and content providers.
- The permissions that are required by the application to access the protected parts of the system and other apps.
- The features required by the app, that affect which devices can install the app from Google Play. These features include both hardware and software features.
- It also specifies the application metadata, which includes the icon, version number, themes, etc.

The android manifest.xml file generally looks like the following:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
   package="com.example.myfirstapplication">

   <application
       android:allowBackup="true"
       android:icon="@mipmap/ic_launcher"
       android:label="@string/app_name"
       android:roundIcon="@mipmap/ic_launcher_round"
       android:supportsRtl="true"
       android:theme="@style/AppTheme">
       <activity android:name=".MainActivity">
           <intent-filter>
               <action android:name="android.intent.action.MAIN" />
               <category android:name="android.intent.category.LAUNCHER" />
           </intent-filter>
       </activity>
       <meta-data
           android:name="preloaded_fonts"
           android:resource="@array/preloaded_fonts" />
   </application>

</manifest>
```

Element Tags of Manifest File

Following are the essential element tags of Manifest.xml files:

1.<manifest>
It is the root element of this element. It consists of a package attribute package that tells the activity's package name.

2.<application>
It is the subelement of the manifest file that includes the declaration of the namespace. It contains certain attributes. These attributes declare the application components, and these attributes include:

- icon
- allowBackup
- label
- theme

3.<activity>
Activity is a subelement of application. It has the declaration of the activity that must be there in the manifest file. It also has certain attributes like name, label, theme, etc.

4.<intent-filter>
It is an element in the activity, it describes the type of intent in which the Android components can respond.

5.<action>
This element provides an action for the intent filter. Each intent filter must have at least one action element in it.

6.<category>
This element adds the category name in an intent-filter.

7.<service>
This element contains the operations that are provided by libraries or APIs.

Manifest File Application Property Elements

Following is a list of important Application Property Elements in manifest.xml (Sub- Node Elements)

1. <uses-permission>: This element specifies the Android Manifest permissions that are requested for the purpose of security.

2. <permission>: This element sets permission to provide access to control for some components of the app.

3. <permission-groups>: This element sets permission to provide access to control for a set of components of the app.

4. <permission-tree>: This element refers to a specific component that is the owner of the set of components.

5. <instrumentation>: This element tells the interaction between the app and the system.

6. <uses-sdk>: This one specifies the compatibility of the app.

7. <uses-configuration>: This element specifies the permissions that are requested for the purpose of security.

8. <uses-feature>: This element specifies one hardware or software feature that is required by the Application.

9. <supports-screen, compatible-screen>: These elements tell the screen size and configuration.

How to set Manifest File Permissions in Android?

Now in this part of the tutorial, we'll see how to set permission in a Manifest file.

An android application must get some permissions to get access to other apps or the Internet. While we build our app, the manifest file gets automatically generated

as manifest.xml. This manifest file contains permissions that are configured by us. A few permissions are applied by default if there is no permission provided by us.

These are written in the manifest file as:

```
<manifest >
<uses-permission .../>
...
<manifest >
```

The permission mentioned in the manifest file can be either granted or rejected by the users. Once the user grants permission to the app, the app can use the protected features. If the user denies permission the app doesn't get permission to access the protected features.

The following permissions are added by default and set to TRUE:

- INTERNET
- ACCESS_NETWORK_STATE
- READ_PHONE_STATE

There are many permissions that are set to FALSE by default but can be set to TRUE as per requirements. A few of these permissions are as follows:

ACCESS_WIFI_STATE	AUTHENTICATE_ACCOUNT	BLUETOOTH	BATTERY_STATS
BIND_APPWIDGET	BROADCAST_WAP_PUSH	BROADCAST_STICKY	BIND_INPUT_METHOD
CALL_PHONE	CHANGE_CONFIGURATION	CAMERA	CLEAR_APP_DATA
CHANGE_WIFI_STATE	CLEAR_APP_USER_DATA	DEVICE_POWER	DELETE_CACHE_FILES
DISABLE_KEYGUARD	DELETE_PACKAGES	EXPAND_STATUS_BAR	EXPAND_STATUS_BAR
FACTORY_TEST	GET_PACKAGE_SIZE	FLASHLIGHT	GLOBAL_SEARCH
HARDWARE_TEST	INTERNAL_SYSTEM_PROCESSES	USE_CREDENTIALS	MANAGE_ACCOUNTS
MANAGE_APP_TOKENS	MODIFY_AUDIO_SETTINGS	MODIFY_PHONE_STATE	NFC
SEND_SMS	PROCESS_OUTGOING_CALLS	SET_ALARM	SET_ALWAYS_FINISH
READ_CALENDAR	KILL_BACKGROUND_PROCESSES	SET_WALLPAPER	VIBRATE
WAKE_LOCK	WRITE_APN_SETTINGS	WRITE_CALENDAR	WARITE_SETTINGS